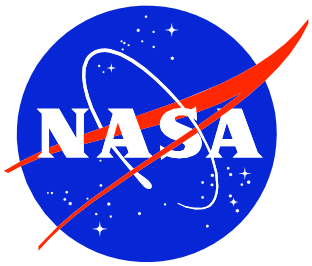
Gamma-Ray Large Area Space Telescope (GLAST) Project

MOC Test & Verification Plan

Version 0.03
August 18, 2004



GODDARD SPACE FLIGHT CENTER
GREENBELT, MARYLAND

Gamma-Ray Large Area
Space Telescope
(GLAST)
Project

MOC Test and Verification Plan

August 18, 2004

Gamma-Ray Large Area Space Telescope (GLAST) Project

MOC Test and Verification Plan

Prepared by:

Jonathan DeGumbia Date
MOC Systems Engineer

Concurrence:

Dennis Small Date
GLAST MOC Lead

Doug Spiegel Date
GLAST MOC Manager

George Sauble Date
GLAST MOC Program Director

Approved by:

Ken Lehtonen Date
GLAST Ground System/Ops Manager

REVISION STATUS

VERSION	DATE	CHANGED BY	DESCRIPTION
v0.01	07/13/04	J. DeGumbia	Initial draft
v0.02	07/30/04	J. DeGumbia	Update
v0.03	08/16/04	J. DeGumbia	Update

Table of Contents

Table of Figures

Table of Tables

Acronyms

COP-1	Communications Operation Procedure-1
COTS	Commercial-Off-The-Shelf
DR	Discrepancy Report
FOT	Flight Operations Team
GLAST	Gamma-ray Large Area Space Telescope
GOTS	Government-Off-The-Shelf
GPS	Global Positioning System
GSFC	Goddard Space Flight Center
GSRD	Ground System Requirements Document
ITOS	Integrated Test and Operation System
I&T	Integration and Test
MOC	Mission Operations Center
MPS	Mission Planning System
NASA	National Aeronautics and Space Administration
PSS	Portable Spacecraft Simulator
SDF	Software Development Folder
STK	Satellite Toolkit
STOL	Systems Test and Operations Language
TBS	To be Supplied
TDRSS	Tracking and Data Relay Satellite System
USN	Universal Space Network

1.0 INTRODUCTION

1.1 PURPOSE

The purpose of this document is to define the testing activities used to validate that the National Aeronautics and Space Administration (NASA) Gamma-ray Large Area Space Telescope (GLAST) Mission Operations Center (MOC) complies with the MOC element level 4 requirements as defined in the *GLAST MOC Functional and Performance Requirements Document* and provides a suitable operational center to help ensure the success of the GLAST mission.

1.2 SCOPE

This document details the test-related plans and activities necessary to test the functionality and performance of the MOC. The scope of the testing covered by this document is limited to all test activities leading up to and including those required to validate the ability of the MOC to satisfy the MOC element level 4 requirements as defined in the *GLAST MOC Functional and Performance Requirements Document*.

Although validation of the MOC level 4 requirements is a necessary part of the verification of the level 3 *GLAST Ground System Requirements Document* (GSRD), the test activities necessary to validate the level 3 requirements are not included in this document. Such plans are detailed in the *GLAST Ground System Test Plan*.

1.3 APPLICABLE DOCUMENTS

The following documents are referenced within this document. The reader is encouraged to use present and future versions of these documents for further research.

- 492-MOC-002*: GLAST MOC Functional and Performance Requirements Document*.
- 433-XXXX-XXX: *GLAST Ground System Test Plan*.
- 433-RQMT-0006: *GLAST Project Ground System Requirements Document*.

2.0 TEST MANAGEMENT

2.1 TEST TEAM ROLES AND RESPONSIBILITIES

Testing of the MOC is the responsibility of the Goldbelt Orca/Omitron MOC development and Flight Operations Teams (FOT); there is not a separate test team for GLAST. The team consists of systems engineers, software developers, and the FOT. Management of the MOC test program is the responsibility of the Lead Ground System Engineer. The system engineers have responsibility for system and acceptance testing of the MOC to verify that all requirements are met and that the system will satisfy flight operations needs.

2.2 TEST ENVIRONMENT

MOC testing will initially be performed at the Omitron development facility. The development lab contains the MOC network, hardware, and software that simulate the MOC facility at Goddard Space Flight Center (GSFC). In addition, the development lab contains the Portable Spacecraft Simulator (PSS), which is used as a test source to simulate telemetry and to verify commands.

The MOC system will also be tested at the Spectrum Astro observatory integration and test (I&T) facility located in Gilbert, AZ. A subset of the MOC functionality (command generation and core telemetry & command capability) will be tested using the spacecraft Hotbench simulator and the actual observatory. Note that MOC testing is not a necessary part of the spacecraft or observatory I&T program.

Testing will also be performed at the MOC facility located in building 14 of GSFC. This testing verifies that the system and its interfaces to external entities perform properly in its operational environment.

3.0 TEST METHODOLOGY

This section describes the methodology for verification of the MOC, including the test approach, planning and preparation, test execution, and reporting.

3.1 TEST APPROACH

Testing will be guided by these principles:

- Testing is driven by defined requirements
- Informal testing during development compliments formal testing prior to each release
- Regression testing is performed for software upgrades
- Tests are conducted by personnel other than those involved in the software development
- Formal testing is conducted using approved procedures with clearly delineated expected results
- Test results are documented

The MOC is being developed, tested, and released in a series of builds to incrementally advance the total system functionality. For each MOC build, the developers perform unit testing for all new software and for modified reused software. Product testing is performed for all commercial-off-the-shelf (COTS) and government-off-the-shelf (GOTS) software packages to ensure the package is installed and configured properly and that all needed functions perform correctly. Following unit and product testing, the components for the build are integrated. During integration, informal testing of component strings may be performed to reduce the reliance on higher level tests to identify integration difficulties. System testing is then conducted to verify that all components have been properly integrated to meet the requirements for that build. Finally, acceptance testing is executed as the run-for-record tests to verify the build is ready for delivery. Reference the *MOC Build Plan* for definition of the functionality and software components designated for each build.

3.1.1 Unit Testing

Unit testing is conducted for each module of new or modified software according to a unit test plan. The unit test plan is an informal plan generated by the developer that defines the purpose of the module and the specific paths and variations to be tested. The unit test plan is reviewed as part of the code walkthrough prior to execution. The test plan and results are documented in the Software Development Folder (SDF) for that particular software component.

3.1.2 Product Testing

Product testing is performed for all COTS and GOTS packages to verify that the configuration setup is correct and that the product performs its necessary functions properly. COTS and unmodified GOTS packages are not tested at the unit level, nor does this testing cover the entire set of functionality that the product may provide. The focus is to verify that the product will perform the functions necessary to satisfy the MOC requirements. Product testing is performed after the installation and configuration setup is complete, according to an informal product test plan that defines the specific test cases to be executed. The test plan and results are documented in the SDF for that particular software component.

3.1.3 Integration Testing

Once the necessary unit and product tests are completed, the associated software components are integrated together to form a software build. Informal testing is performed throughout integration to verify that the interfaces between the various software components are working correctly and that the integrated components produce the necessary results for particular system thread.

3.1.4 System Testing

System testing begins after the successful integration of a build. The system tests are executed according to the system test procedures defined for that build. System test cases are first developed and reviewed in a walkthrough. Detailed test procedures are then developed to provide a guide for testers, provide a record of test steps, and ensure repeatability. Test procedure versioning is tracked and controlled using the Docushare document management product. The tester executes the test procedure, documents the results in the test form, and assesses the results to ensure compliance with requirements. Discrepancy Reports (DRs) are written for any discrepancies identified. The MOC Manager assesses the DRs to determine if corrections are required for the current build or deferred to a subsequent build or patch release.

3.1.5 Acceptance Testing

Acceptance testing is the final step in verification of the build. The testers perform acceptance testing to verify the functionality and performance of the MOC system and to verify that all requirements are met. Acceptance testing is performed at the MOC development lab. The testers execute the system test cases and formally record the results for each case. Upon successful execution and verification of the test results, the associated requirements are checked off in the MOC verification matrix. All discrepancies encountered are documented in DRs and are assessed by the MOC development team to determine if the delivery of the build should proceed or if corrective action needs to be taken prior to delivery.

3.1.6 Regression Testing

Regression testing will be performed for each successive build, for patch releases, and for new MOC installations. Regression tests ensure that functionality from the previous build has been retained. A suite of regression tests is derived from system test cases generated for the builds. This test suite will grow as subsequent builds are completed. Regression testing is performed at the GLAST MOC.

3.1.7 Testing of External Systems

The MOC relies on external software systems as part of the tool set necessary to accomplish GLAST mission needs. For those external systems that are used directly by MOC operations personnel (not data interfaces), the system must be tested by the MOC. This includes the Space Network Web-based Scheduling Interface (SWSI) and the NASA Orbital Information Group web site. Testing of these external systems will be treated similarly as COTS/GOTS products, in that an informal test plan will be generated and executed to verify the MOC's ability to accomplish the needed functions from the external system. Discrepancies will be captured in DRs and their resolution will be coordinated through direct contact. Testing of all external interface systems is included in the system and acceptance testing, as part of data flow tests, and tests exercising complete system threads. The test plan and results are documented in the SDF.

3.2 PLANNING AND PREPARATION

Planning for the test program includes identification of required test cases, identification of test resources, scheduling test preparation, and executing test tasks. Test preparation includes development of test case descriptions, detailed procedures, expected results/acceptance criteria, and generation of test data. Test case descriptions are documented using a standard template, shown in Figure 1, indicating test purpose, configuration, resources, procedure, and acceptance criteria. The test case descriptions are included in Appendix A of this document, which will be updated with the appropriate test cases prior to each MOC build.

Test Case ID:
Title:

Test Objectives: Identify the functional capabilities being exercised.

Requirements to be Verified: List MOC requirements to be verified by the test.

Test Configuration: Provide a block diagram showing the major processing elements, data flows, and data communication mechanisms.

Hardware/Software Configuration: Provide release number and/or version numbers of all relevant hardware and software used during the test.

Participants and Support Requirements: Identify the participatin g organizations and equipment, circuits, and personnel support required. Identify any test tool requirements in this section.

Test Data: Describe and identify by file name, script name or other designation all required test data sets. Provide the source and physical locations of the data. The description should include volumes and errors or insertion of anomalous conditions.

Test Case Descriptions: Provide a brief narrative description of each test case along with high -level success criteria. Note tha t the goal for test cases is to be modular and focused on certain capabilities or functions, thus providing maximum flexibility to focus on delivered capabilities or those areas of current interest during test execution.

Test Procedures: Describe the test set-up, test execution, and test termination.

Figure 1 Test Case Description Template

3.3 TEST EXECUTION

Tests are executed following the approved test procedure for each test case. The tester first verifies the correct software and hardware configuration and presence of appropriate test data.

Any required test sources (e.g., PSS) are configured and set up for the run. The test is then executed according to the procedure, and the results are documented for each relevant step. The tester verifies that all products were generated correctly and that all relevant data is saved in the appropriate directory. Upon the completion of the test, the results are assessed and the requirements are verified. All associated test data is archived in the appropriate test directory.

3.4 REPORTING

Test summary reports are generated to document the results of each formal test. Details of the test results are documented in test notebooks and are made available for review. Test notebooks will contain:

- Test Procedure Title and date/version
- Test conductor name
- Date/time of test
- Test configuration information
- Test data used
- Description of anomalies and DR numbers, as appropriate
- Location of resulting test data

After acceptance testing for each build is completed, the MOC verification matrix is edited to indicate whether the requirement has been completely verified, partially verified, or requires retest (pending subsequent build or patch release).

DRs are prepared to record discrepancies discovered during conduct of formal and selected informal testing. The MOC Manager works with the MOC team to determine the course of action to address each discrepancy. Actions may include additional testing and analysis, changes to procedures, recommendations for hardware or software changes, work-arounds, etc. Actions taken and the results are recorded on the DR form. DRs are closed upon approval by the MOC Manager.

Regular briefings of MOC status, including significant open DR's and DR metrics, are made to GLAST project management on a regular basis. DR's are maintained in the web-based TRACKGEAR system, and are accessible for Project-level review at any time.

4.0 TEST DESCRIPTIONS

This section provides a description of the tests that form the MOC test set and a listing of test cases by build. A test case identifier is used to cross-reference the test case descriptions contained in Appendix A. Refer to the *MOC Build Plan* for definition of the build contents.

The MOC test suite consists of a set of tests for each functional area or major software component, as well as overall system-level tests to verify major system threads and end-to-end capability. The test sets are summarized below:

- ITOS:
  - Verifies telemetry receipt, processing, distribution, display, monitoring, quick-look level 0 product generation, and archiving.
  - Verifies Command processing, COP-1 verification, STOL execution.
- STK:
  - Verifies predictive orbital ephemeris from observatory GPS data
  - Verifies attitude-dependent access times for TDRSS and USN contacts
- FASAT:
  - Verifies automatic paging notification
- TRACKGEAR:
  - Verifies DR logging and tracking
- Mission planning & scheduling:
  - STK/Scheduler: Verifies schedule deconfliction
  - MPS: Verifies contact scheduling, orbital product generation, and command load generation
- Monitoring:
  - Verifies as-flown timeline generation
- Web & Remote Access:
  - Verifies ability to access products via the web
- System level tests:
  - Verifies data flows to external interfaces
  - Verifies 72 hour continuous operations

Selected test cases will be executed for each MOC Build, as defined in the tables below.

Table 1 Build 1 Test Cases

Package/ Component	Test Case ID	Description

Table 2 Build 2 Test Cases

Package/ Component	Test Case ID	Description

Table 3 Build 3 Test Cases

Package/ Component	Test Case ID	Description

APPENDIX A – Test Case Descriptions

TBS